IN THE CLAIMS:

Please amend the claims as follows. The claims are in the format as required by 35 C.F.R. § 1.121.

1. (Currently amended) A method of modeling an arbitrarily complex environment, comprising:

On a computer having a computer memory and a processor, defining data structures for dynamically accommodating changes to the arbitrarily complex environment in a data model, wherein the data structures comprise components and relationships;

representing ~~at least two entities~~ each atomic entity in the arbitrarily complex environment~~, wherein each entity is represented~~ with a component in the data model, wherein each atomic entity is a logical or physical entity in the arbitrarily complex environment and wherein the component has a set of fields which contain information relating to the atomic entity associated with the component;

representing an association or a dependency between ~~the at least~~ two or more components in the data model with a relationship, wherein the relationship has a name field containing a relationship name which is built programmatically and which automatically associates the relationship with two or more physical or logical entities that correspond to the two or more components;

~~defining a hierarchy of the components and relationships; and~~

~~querying according to specific search criteria~~; and

automatically changing the relationship to reflect a corresponding change in the arbitrarily complex environment.

2. (Currently amended) The method of claim 1, wherein each component is instantiated based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, a property, and the presence of events, wherein the name field associates the component with a particular atomic entity.

3. (Currently amended) The method of claim 1 **[[2]]**, wherein each component type is in a hierarchy of component types.

4. (Currently amended) The method of claim 2 **[[3]]**, wherein each property has a data type of one or more of a string, a numeric, a Boolean, a link, a date/time and a custom type ~~component type is a parent type or a subtype~~.

5. (Currently amended) The method of claim 2 **[[4]]**, wherein each property is a data structure having a name, a description and a value ~~the hierarchy of component types is tailored to the environment~~.

6. (Currently amended) The method of claim 2, wherein a component is related to a set of checks, wherein each check is a piece of logic which performs operations ~~each relationship is instantiated based on a relationship type~~.

7. (Currently amended) The method of claim 6, wherein ~~each relationship type is in a hierarchy of relationship types~~ a check includes an operation for checking the status of a relationship.

8. (Currently amended) The method of claim 1 **[[7]]**, wherein each relationship type is a parent type or a subtype.

9. (Currently amended) The method of claim 1 **[[8]]**, wherein ~~the hierarchy of~~ each relationship ~~types is tailored to the environment~~ comprises a name field, a description field, a property field, and a checks field.

10. (Original) The method of claim 6, wherein each component is represented in a component table.

11. (Original) The method of claim 10, wherein each component type is represented in component type table.

12. (Original) The method of claim 11, wherein each relationship is represented in a relationship table.

13. (Original) The method of claim 12, wherein each relationship type is represented in relationship type table.

14. (Original) The method of claim 13, wherein the relationship table links each relationship to at least two components.

15. (Currently amended) The method of claim 14, wherein the relationship table and the relationship type table are distinct.

16. (Currently amended) A system for modeling an arbitrarily complex environment, comprising:

a ~~general purpose~~ computer having a memory for storing a set of computer-executable instructions and a processor for executing the computer-executable instructions operable to:

~~a software product, wherein the software product is on a computer readable medium and is capable of instructing a general purpose computer to:~~

define data structures for dynamically accommodating changes to the arbitrarily complex environment in a data model, wherein the data structures comprise components and relationships;

generate ~~and save~~ a set of components in the data model, wherein ~~each~~ the set of components represents **[[**an**]]** each atomic entity within the arbitrarily complex environment, wherein each atomic entity is a logical or physical entity in the arbitrarily complex environment and wherein each component has a set of fields which contain information relating to the atomic entity associated with each component;

generate ~~and save~~ a set of relationships in the data model, wherein each relationship represents an association between at least two of the components, wherein the relationship has a name field containing a relationship name which is built programmatically and which automatically associates the relationship with two or more physical or logical entities that correspond to the two or more components;

~~define a hierarchy of the set of components and the set of relationships;~~

and

automatically change the relationship to reflect a corresponding change in the arbitrarily complex environment

~~query according to specific search criteria~~.

17. (Currently amended) The system of claim 16, wherein each component is instantiated based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, a property, and the presence of events, wherein the name field associates the component with a particular atomic entity.

18. (Currently amended) The system of claim 16 [[17]], wherein each component type is in a hierarchy of component types.

19. (Currently amended) The system of claim 18, wherein each property has a data type of one or more of a string, a numeric, a Boolean, a link, a date/time and a custom type ~~component type is a parent type or a subtype~~.

20. (Currently amended) The system of claim 19, wherein each property is a data structure having a name, a description and a value ~~the hierarchy of component types is tailored to the environment~~.

21. (Currently amended) The system of claim 15, wherein a component is related to a set of checks, wherein each check is a piece of logic which performs operations ~~each relationship is instantiated based on a relationship type~~.

22. (Currently amended) The system of claim 21, wherein a check includes an operation for checking the status of a relationship ~~each relationship type is in a hierarchy of relationship types~~.

23. (Original) The system of claim 22, wherein each relationship type is a parent type or a subtype.

24. (Currently amended) The system of claim 16 [[23]], wherein ~~the hierarchy of~~ each relationship ~~types is tailored to the environment~~ comprises a name field, a description field, a property field, and a checks field.

25. (Original) The system of claim 21, wherein each component is represented in a component table.

26. (Original) The system of claim 25, wherein each component type is represented in component type table.

27. (Original) The system of claim 26, wherein each relationship is represented in a relationship table.

28. (Original) The system of claim 27, wherein each relationship type is represented in relationship type table.

29. (Original) The system of claim 28, wherein the relationship table links each relationship to at least two components.

30. (Currently amended) The system of claim 29, wherein the relationship table and the relationship type table are distinct.

31. (Currently amended) A software product capable of instructing a ~~general purpose~~ computer on a computer-readable medium, wherein the computer has a computer memory and a processor, wherein the software product comprises:

an instruction to define data structures for dynamically accommodating changes to an arbitrarily complex environment in a data model, wherein the data structures comprise components and relationships;

an instruction to represent ~~at least two entities~~ each atomic entity in the arbitrarily complex environment, wherein each atomic entity is a logical or physical entity in the arbitrarily complex environment and wherein ~~represented with a component and~~ the component ~~is saved~~ has a set of fields which contain information relating to the atomic entity associated with the component;

an instruction to represent an association or a dependency between ~~the at least~~ two or more components with a relationship, wherein the relationship has a name field containing a relationship name which is built programmatically and which automatically associates the relationship with two or more physical or logical entities that correspond to the two or more components;

~~an instruction to define a hierarchy of the components and relationships~~; and

an instruction to automatically change the relationship to reflect a corresponding change in the arbitrarily complex environment

~~query according to specific search criteria~~.

32. (Currently amended) The software product of claim 31, wherein each component is instantiated based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, a property, and the presence of events, wherein the name field associates the component with a particular atomic entity.

33. (Currently amended) The software product of claim 31 **[[**32**]]**, wherein each component type is in a hierarchy of component types.

34. (Currently amended) The software product of claim 33, wherein each property has a data type of one or more of a string, a numeric, a Boolean, a link, a date/time and a custom type ~~component type is a parent type or a subtype~~.

35. (Currently amended) The software product of claim 34, wherein each property is a data structure having a name, a description and a value ~~the hierarchy of component types is tailored to the environment~~.

36. (Currently amended) The software product of claim 32, wherein a component is related to a set of checks, wherein each check is a piece of logic which performs operations ~~each relationship is instantiated based on a relationship type~~.

37. (Currently amended) The software product of claim 36, wherein a check includes an operation for checking the status of a relationship ~~each relationship type is in a hierarchy of relationship types~~.

38. (Previously Presented) The software product of claim 37, wherein each relationship type is a parent type or a subtype.

39. (Currently amended) The software product of claim 31 **[[38]]**, wherein ~~the hierarchy of~~ each relationship ~~types is tailored to the environment~~ comprises a name field, a description field, a property field, and a checks field.

40. (Previously Presented) The software product of claim 6, wherein each component is represented in a component table.

41. (Previously Presented) The software product of claim 40, wherein each component type is represented in component type table.

42. (Previously Presented) The software product of claim 41, wherein each relationship is represented in a relationship table.

43. (Previously Presented) The software product of claim 42, wherein each relationship type is represented in relationship type table.

44. (Previously Presented) The software product claim 43, wherein the relationship table links each relationship to at least two components.

45. (Currently amended) The software product of claim 44, wherein the relationship table and the relationship type table are distinct.

46. (Previously Amended) The method of claim1, further comprising, utilizing a typing system to define the hierarchy of components and relationships.

47. (Previously Amended) The method of claim 46, wherein the typing system further includes a generic model structure to define a hierarchy of components and relationships.

48. (Previously Amended) The method of claim 47, wherein a data structure is associated with the generic data model.

49. (Currently Amended) The method of claim 48, wherein the data structure **[[**is**]]** associated with the generic data model is stored utilizing a table schema.

50. (Currently Amended) The method of claim 49, wherein the table schema does not change with an addition of a ~~Previously Amended~~ data structure or types of data structures.

51. (Previously Amended) The method of claim 1, wherein each of the relationships or components has a type, wherein the type is a category of the relationships or components and wherein the relationships or components type has the same properties.

52. (Previously Amended) The method of claim 51, wherein the relationships or components has different values for the same properties associated with the type.

53. (Previously Amended) The method of claim 52, wherein the relationship or component type further includes a subtype, wherein the subtype inherits all the properties of the relationship or component type.